Genetic Algorithms :

Mayer, Alexandre

Published in:
Vietnam Journal of Science and Technology

DOI:
10.15625/2525-2518/55/6a/12361

Publication date:
2017

Document Version
Publisher's PDF, also known as Version of record

Vietnam Journal of Science and Technology ***55*** *(6A) (2017) 9-17*

GENETIC ALGORITHMS: AN EVOLUTIONARY APPROACH TO OPTICAL ENGINEERING

Alexandre Mayer

Department of Physics, University of Namur, Rue de Bruxelles 61, Namur, Belgium

Email*: alexandre.mayer@unamur.be*

Received: 15 July 2017; Accepted for publication: 15 December 2017

ABSTRACT

We present a Genetic Algorithm that we developed to address optimization problems in optical engineering. Our objective is to determine the global optimum of a problem ideally by a single run of the genetic algorithm. We want also to achieve this objective with a reasonable use of computational resources. In order to accelerate the convergence of the algorithm, we establish generation after generation a quadratic approximation of the fitness in the close neighborhood of the best-so-far individual. We then inject in the population an individual that corresponds to the optimum of this approximation. We also use randomly-shifted Gray codes when applying mutations in order to achieve a better exploration of the parameter space. We provide automatic settings for the technical parameters of our algorithm and apply it to typical benchmark problems in 5, 10 and 20 dimensions. We show that the global optimum of these problems can be determined with a probability of success in one run of the order of 95-97 % and an average number of fitness evaluations of the order of 400-750×n, where n refers to the number of parameters to determine. We finally comment some applications of this algorithm to real-world engineering problems.

Keywords: Genetic Algorithms, optical engineering, Randomly-Shifted Gray Codes, quadratic approximation, singular value decomposition.

1. INTRODUCTION

With Genetic Algorithms, we mimic natural selection in order to determine the global optimum of complex problems [1-3]. The idea consists in working with a virtual population of individuals, each individual being representative of a given set of physical parameters for the problem we seek at optimizing. We start with a random population. The best individuals are then selected. They generate new individuals for the next generation by crossing the parameters of the selected individuals. Random mutations in the coding of parameters are finally introduced. This strategy is applied from generation to generation until the population converges to an optimum of the problem considered.

This approach is especially suited to parallel computing since the evaluation of the different solutions represented in the population can be done independently. For problems that are

computationally expensive, it is desirable to determine the global optimum ideally in a single run. It is also desirable to accelerate the refinement of the solution, once the good spot has been identified. Although progress is generally fast in the first generations, genetic algorithms are slower at finalizing the optimization. This problem is usually addressed by coupling the genetic algorithm with a local optimizer (Memetic Algorithms) [4]. This local optimizer requires however an extra budget of fitness evaluations. A second approach consists in analyzing the data already collected by the algorithm. By establishing quadratic approximations of the function considered, we can indeed guide the algorithm to promising directions and accelerate the refinement of the solution without increasing the number of evaluations [5-8].

We present in this article a genetic algorithm that we developed to address computationally expensive optimization problems encountered in the engineering of optical devices. Our objective was to determine the global optimum of this type of problems ideally in a single run. We want also to achieve this objective with a reasonable use of computational resources. The details of this algorithm are presented in the Methodology section. We then apply this algorithm to typical benchmark problems in 5, 10 and 20 dimensions in order to demonstrate its performance. We finally comment some applications of this algorithm to optical engineering problems.

2. MATERIALS AND METHODS

Let $f = f(x_1, \ldots x_n)$ be a function a n variables x_i to be optimized, where $x_i \in [x_i^{\min}, x_i^{\max}]$ with a specified granularity of Δx_i. We work with a virtual population of n_{pop} individuals. Each individual is representative of a given set of parameters $\{x_i\}_{i=1,n}$. These parameters are represented by a string of binary digits ("DNA"), which consists of n "genes". The x_i value encoded by each gene is given by $x_i = x_i^{\min} + \langle gene\ i \rangle . \Delta x_i$, where $\langle gene\ i \rangle \in [0, 2^{n(i)}-1]$ refers to the binary value of the gene i and $n(i)$ stands for the number of bits in this gene. We use the Gray code instead of standard binary for this encoding of parameters [9 - 10]. $n_{\text{bits}} = \Sigma_i\, n(i)$ will refer to the total number of bits in a DNA.

We start with a random population and proceed iteratively through the following steps. We first compute the "fitness" $f(x_1, \ldots x_n)$ of each individual. The population is then sorted according to this fitness. We replace the worst n_{rand} individuals of the population by random individuals for the next generation ($n_{\text{rand}} = 0.1 \times n_{\text{pop}} \times (1-p)$, with $p = |s-0.5|/0.5$ a progress indicator and s the genetic similarity). The genetic similarity s corresponds to the fraction of the bits in the population whose value is identical to the best individual [9, 11]. We then select $N = n_{\text{pop}} - n_{\text{rand}}$ individuals in the remaining part of the population by a *rank-based roulette wheel selection* (the best individuals have more chance to be selected; the selection probabilities decrease linearly with the ranking) [2]. For two "parents" selected, we define two "children" for the next generation. They are obtained either (i) by a one-point crossover of the parents' DNA (probability of 70 %), or (ii) by a simple replication of the parents (probability of 30 %). The children obtained by crossing the parents' DNA are subjected to mutations. Each bit of their DNA has a probability $m = 0.95/n_{\text{bits}}$ to be reversed. These mutations are actually applied on randomly shifted versions of the original Gray code: the bit content of each gene is first expressed in a randomly-shifted version of the original Gray code; mutations are then applied to this modified encoding; the result is finally translated back to the original encoding. The shift considered for each gene is the same for all individuals in the population. It is reset randomly every generation. This application of mutations to a changing encoding scheme turns out to improve the exploration of the parameter space [12-14].

The data collected by the genetic algorithm is analyzed generation after generation in an attempt to infer the final solution. The idea consists in establishing a quadratic approximation of the fitness in the close neighborhood of the best-so-far individual. We then inject in the population an individual that corresponds to the optimum of this quadratic approximation [5-8]. The details of this method are given in the Appendix. We finally apply elitism to make sure that the best solution is never lost when going from one generation to the next. These steps of selection, crossover, mutation and the analysis of collected data are repeated from generation to generation until a convergence criterion is met

3. RESULTS AND DISCUSSION

We apply in this section our algorithm to typical benchmark problems in 5, 10 and 20 dimensions in order to demonstrate its performance. We are interested by a class of problems in which each parameter x_i has a specified granularity Δx_i such that $(x_i^{max}-x_i^{min})/\Delta x_i \sim 1000$. The number of bits required for the representation of the parameters $\{x_i\}_{i=1,n}$ is therefore of the order of $n_{bits} \sim 10 \times n$, with n the dimension of the problem.

Table 1. Test functions used for the benchmarking. The conventional name of some of these functions is as follows: Sphere (#1), Rotated Hyper-Ellipsoid (#2), Rosenbrock (#3), Schwefel F7 (#6), Levy (#7), Rastrigin (#8), Ackley (#9) and Griewank (#10).

#	Formula	$[x_i^{\min}, x_i^{\max}, \Delta x_i]$
1	$f(\vec{x}) = \sum_{i=1}^{n} x_i^2$	$[-5.12, 5.12, 0.01]$
2	$f(\vec{x}) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	$[-65.5, 65.5, 0.1]$
3	$f(\vec{x}) = \sum_{i=1}^{n-1} [100\,(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$	$[-2.05, 2.05, 0.0025]$
4	$f(\vec{x}) = n(x_1 - 1)^2 + \sum_{i=2}^{n} (2x_i^2 - x_{i-1})^2$	$[0, 10, 0.0025]$
5	$f(\vec{x}) = -\Pi_{i=1}^{n} \cos(x_i)^2 \exp(-x_i^2/10)$	$[-5, 5, 0.01]$
6	$f(\vec{x}) = -\sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500, 1]$
7	$f(\vec{x}) = \sin^2(\pi w_1) + \sum_{i=1}^{n-1} (w_i - 1)^2 [1 + 10 \sin^2(\pi w_i + 1)]$ $+(w_n - 1)^2 [1 + \sin^2(2\pi w_n)],\ w_i = 1 + (x_i - 1)/4$	$[-10, 10, 0.01]$
8	$f(\vec{x}) = 10\,n + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$	$[-5.12, 5.12, 0.01]$
9	$f(\vec{x}) = -a \exp\left(-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(cx_i)\right)$ $+a + e,\ a = 20, b = 0.2, c = 2\pi$	$[-32.8, 32.8, 0.025]$
10	$f(\vec{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(x_i/\sqrt{i})$	$[-600, 600, 0.25]$

The ten analytical functions considered for this benchmarking are given in Table 1. The table provides the boundaries $[x_i^{min}, x_i^{max}]$ as well as the granularities Δx_i used for each function. The objective of the genetic algorithm is ideally to determine the global minimum (f^*_{opt}) of all these functions. We consider in practice that a target Δf_{target} is reached by a given run of the genetic algorithm if $|f(\vec{x}_{best}) - f^*_{opt}| \leq \Delta f_{target}$, where $f(\vec{x}_{best})$ is the best-so-far solution found by the genetic algorithm. We then measure the probability $P(\Delta f_{target})$ that specific targets of 10^{-1}, 10^{-2}, 10^{-3} and 10^{-4} are reached by a single run of the genetic algorithm. This quantity is given by

$P(\Delta f_{target})$=#success/#run, where #success stands for the number of runs in which the target was reached and #run stands for the total number of runs. We also measure the number of fitness evaluations required on average to reach a given target. This quantity is given by $\langle n_{eval} \rangle (\Delta f_{target})$=#eval_target_not_reached/#run, where #eval_target_not_reached is the number of fitness evaluations in all generations for which the target was not reached [15].

Our objective is to have a probability $P(\Delta f_{target})$ as high as possible to reach a target accuracy Δf_{target} of 10^{-4}, while keeping $\langle n_{eval} \rangle$ of the order of $1000 \times n$, where n is the number of parameters to determine. This is indeed the budget of fitness evaluations with which genetic algorithms start being comfortable. We provide in this article a set of technical parameters that enable the genetic algorithm to work with a reduced number of fitness evaluations. We chose for this economy purpose a population size n_{pop} of 50 individuals only, for problems in 5, 10 and 20 dimensions. We allow a maximum of $30 \times n_{bits}$ generations per run. The genetic algorithm stops however if there is no improvement in the best fitness after $1.5 \times n_{bits}$ generations or if the total number of fitness evaluations exceeds $10{,}000 \times n$ in a given run. We also stop the genetic algorithm if the genetic similarity $s \geq 1\text{-}m$, with m the mutation rate, or if the mean value $\langle s \rangle$ of the genetic similarity over the last $1.5 \times n_{bits}$ generations is higher than $1\text{-}3 \times m$. As for problems that are computationally expensive, we keep a record with all fitness evaluations in order to avoid evaluating several times the same set of parameters in a given run of the genetic algorithm.

The benchmark results are summarized in Table 2. It turns out that a target accuracy of 10^{-4} on the global optimum of the functions considered could be reached with a probability of success in one run of 95.2 % in 5 dimensions, 97.4 % in 10 dimensions and 97.3 % in 20 dimensions. These values account for the ten functions considered in the test set. The target accuracy of 10^{-4} was actually achieved for all functions separately, with excellent success probabilities. The use of randomly shifted Gray codes when applying mutations and the use quadratic approximations of the fitness for inferring the final solution both contribute significantly to these results. The average number of fitness evaluations ($\langle n_{eval} \rangle$) required to reach a Δf_{target} of 10^{-4} was 1,808 in 5 dimensions, 4,314 in 10 dimensions and 14,530 in 20 dimensions. We therefore achieve $\langle n_{eval} \rangle / n$ ratios of 362 in 5 dimensions, 431 in 10 dimensions and 726 in 20 dimensions, thus keeping within our budget of $1000 \times n$ fitness evaluations on average for a given run of the genetic algorithm.

These results compare very well with those provided by the state-of-the-art genetic algorithm CMA-ES (real-valued encoding of parameters) [16]. By applying CMA-ES with its default settings on the test functions considered in this work (see Table 1), the probability to reach the target accuracy Δf_{target} of 10^{-4} in one run is reduced to 60.8% in 5 dimensions, 59 % in 10 dimensions and 59 % in 20 dimensions. The $\langle n_{eval} \rangle / n$ ratios associated with this target accuracy of 10^{-4} are 402 in 5 dimensions, 462 in 10 dimensions, and 535 in 20 dimensions. The budgets of fitness evaluations are comparable. CMA-ES however generally fails at detecting the global optimum of some multi-modal functions considered in our test suite, in particular the test functions #5, #6 (Schwefel), #8 (Rastrigin) and #10 (Griewank). In contrast, the genetic algorithm presented in this work (binary encoding of parameters, with randomly shifted Gray codes when applying mutations) achieves a better exploration of the parameter space, which leads to the detection of the global optimum of these multi-modal functions. Statistical hypothesis testing enables one to reject the null hypothesis H_0 that "the genetic algorithm presented in this work does not achieve a higher probability to reach a Δf_{target} of 10^{-4} in one run, compared to CMA-ES" at a significance level α of 10^{-5}.

Table 2. Benchmark results for problems in 5, 10 and 20 dimensions. These results consist of the probability to reach specific targets in a single run, the average number of fitness evaluations required to reach these targets and the number of functions for which the targets were reached at least once in ten runs. These statistics were generated by running the genetic algorithm 500 times on each test function.

Δf_{target}		$n = 5$	$n = 10$	$n = 20$
10^{-4}	$P(\Delta f_{\text{target}})$	95.2%	97.4%	97.3%
	$\langle n_{\text{eval}} \rangle$	1,808	4,314	14,530
	#fct($P \geq 10\%$)	all	all	all
10^{-3}	$P(\Delta f_{\text{target}})$	98.9%	99.1%	98.7%
	$\langle n_{\text{eval}} \rangle$	1,576	4,044	13,944
	#fct($P \geq 10\%$)	all	all	all
10^{-2}	$P(\Delta f_{\text{target}})$	99.0%	99.3%	99.0%
	$\langle n_{\text{eval}} \rangle$	1,432	3,756	13,313
	#fct($P \geq 10\%$)	all	all	all
10^{-1}	$P(\Delta f_{\text{target}})$	99.2%	99.3%	99.0%
	$\langle n_{\text{eval}} \rangle$	1,146	3,309	12,078
	#fct($P \geq 10\%$)	all	all	all

In previous work [11, 17-18], we used genetic algorithms to address optical engineering problems. In Ref. [11], the objective was to maximize the light-extraction efficiency of a GaN Light-Emitting Diode (LED). The genetic algorithm had to determine the geometrical and material characteristics of a periodic texturation for the surface of the GaN, with the objective to maximize the percentage of light produced within the GaN that is actually transmitted into air. In Ref. 17, we used a multi-objective genetic algorithm to improve the performance of a solar thermal collector that consists of an aluminum waffle-shaped structure, with conformal coatings of $NiCrO_x$ (cermet) and SnO_2 (anti-reflection coating). This problem was characterized by two objectives: (i) maximizing the absorption of solar radiation, and (ii) minimizing the emission of thermal radiation. A multi-objective genetic algorithm provides in this case a list of Pareto-optimal solutions, which represent different compromises between these two objectives. In Ref. 18, we used a genetic algorithm to maximize the absorption of light in a thin film (40 μm) of crystalline silicon for an application in photovoltaics. The device considered in this work was characterized by different layers of materials, with specific purposes (a-Si:H and AlO_x for passivation of the silicon, SiN_x to get an anti-reflection effect and ITO to achieve reflection on the back side of the device). The thickness of these different layers had to be optimized. A periodic array of inverted pyramids was also considered in order to achieve a graded-refractive-index effect (increased light penetration) as well as a light-trapping effect (keeping light trapped in the silicon until it gets absorbed). The geometrical characteristics of these inverted pyramids had also to be optimized in order to achieve these effects.

These different engineering problems were addressed with versions of the genetic algorithm whose main lines are similar to those presented here. The number *n* of parameters to determine was of the order of 5 or 6. Each parameter had a specified granularity such that the number of possible values was of the order of ~1000 as in this work. Each evaluation of the fitness involved typically 24 hours of cpu time. A multi-agent version of the genetic algorithm was

therefore developed in order to do all the fitness calculations of a given generation in parallel and a record of all fitness evaluations was maintained in order to avoid duplicate evaluations. The target accuracy of 10^{-4} on the objective function was relevant to this previous work, where the numerical simulations involved in the fitness calculations had indeed an accuracy limited to this range. The mutation rate m was set to 1% in this previous work, which satisfy a biological constrain that $n_{\text{bits}} \times m < 1$ in order for individuals to have a chance to be unaffected by mutations [19]. The preservation of good solutions in the population enables indeed the genetic algorithm to use them as seeds for establishing better solutions. This biological constrain is addressed here automatically since the mutation rate is settled as $m = 0.95/n_{\text{bits}}$. A crossover rate of 70 % implies a lifetime of three generations for a given individual. This is also consistent with the necessity to keep a reservoir of good solutions in the population. A population size n_{pop} of 100 individuals was finally considered in this previous work. This article shows that working with 50 individuals may be sufficient. This is essentially a consequence of using generation after generation quadratic approximations of the fitness in order to infer the final solution. The use of randomly shifted Gray codes when applying mutations also contributes significantly to this improvement. The application of mutations to a changing encoding scheme helps indeed the genetic algorithm escape local optima. It also enables a coupling between parameter values whose encoding could otherwise be too distant in a given binary representation.

3. CONCLUSIONS

We presented a genetic algorithm that we use for determining the global optimum of functions $f(x_1, \ldots x_n)$ that depend on n physical parameters x_i. We are interested by a class of problems in which each parameter can take of the order of $(x_i^{\max}\text{-}x_i^{\min})/\Delta x_i \sim 1000$ possible values and where a target accuracy Δf_{target} of 10^{-4} on the global optimum is sufficient for the application considered. This is the characteristic situation for the optical engineering problems referred to in this work. Our objective is to determine the global optimum of this type of problems ideally by a single run of the genetic algorithm. We want also to make a reasonable use of computational resources. By establishing generation after generation a quadratic approximation of the fitness in the close neighborhood of the best-so-far individual, it is possible to infer more rapidly the final solution. We can also improve the exploration of the parameter space by using randomly shifted Gray codes when applying mutations. These ideas make it possible to achieve our objectives. We could indeed determine the global optimum of typical benchmark problems in 5, 10 and 20 dimensions with a probability of success in one run of the order of 95-97 % and an average number of fitness evaluations of the order of 400-750×n, where n refers to the dimension of the problem. These results compare very well with those provided by the state-of-the-art algorithm CMA-ES. These developments will certainly be useful for future optimization problems.

Acknowledgements. A. M. is funded as Research Associate by the Fund for Scientific Research (F.R.S.-FNRS) of Belgium. He is member of NaXys, Namur Center for Complex Systems, University of Namur, Belgium. This research used resources of the "Plateforme Technologique de Calcul Intensif (PTCI)" (http://www.ptci.unamur.be) located at the University of Namur, Belgium, which is supported by the F.R.S.-FNRS under the convention No. 2.4520.11. The PTCI is member of the "Consortium des Equipements de Calcul Intensif (CECI)" (http://www.ceci-hpc.be).

Appendix: Analysis of collected data by quadratic approximations of the fitness

In order to accelerate the convergence of the genetic algorithm, we establish generation after generation a quadratic approximation of the fitness in the close neighborhood of the best-so-far solution. We then inject in the population an individual that corresponds to the optimum of this approximation.

The approximation to establish has the form

$$f(\vec{x}) = a_0 + \vec{A}_1.\vec{\mathrm{X}} + \tfrac{1}{2}\vec{\mathrm{X}}.A_2\vec{\mathrm{X}} \qquad (1)$$

where $\vec{\mathrm{X}} = \Delta^{-1}(\vec{x} - \vec{x}_{ref})$ with $\Delta = diag[\Delta_1,...,\Delta_n]$ a diagonal matrix that contains the granularity Δ_i associated with each variable and $\vec{x}_{ref}$ the best-so-far solution found by the genetic algorithm. a_0 is a scalar, $\vec{A}_1$ a vector of size n and A_2 a matrix of size $n \times n$. Since the matrix A_2 is symmetric, there are $N_{coeff}=1+n+n.(n+1)/2$ unknown coefficients in a_0, $\vec{A}_1$ and A_2 to determine. We use for the adjustment of these coefficients all data points collected by the genetic algorithm for which $\max_i |x_i - x_{i,ref}|/\Delta_i \leq W$, where W determines the half-width of the region considered for the selection, in units of Δ_i. We start with a half-width parameter W of 5 steps and increase it progressively (by increments of 2 steps), until at least $2 \times N_{coeff}$ data points are included in this selection.

The coefficients of the quadratic approximation are established by requiring that $\left\| \vec{f} - M\vec{A} \right\|^2$ be minimized, where $\vec{f}$ is a vector that contains the $f(\vec{x})$ values of the N_{select} data points selected and $\vec{A}$ is a vector of size N_{coeff} that contains the unknown coefficients in a_0, $\vec{A}_1$ and A_2. M is a matrix of size $N_{select} \times N_{coeff}$ with coefficients defined from Eq. (1). We use the Singular Value Decomposition of the matrix M and express it as $M=U\Sigma V^t$, where U is an orthonormal matrix of size $N_{select} \times N_{coeff}$ and V an orthonormal matrix of size $N_{coeff} \times N_{coeff}$. Σ is a diagonal matrix of size $N_{coeff} \times N_{coeff}$ that contains the singular values σ_k of the matrix M [20]. The coefficients of the quadratic approximation can then be calculated by $\vec{A} = V\Sigma^{+}U^t\vec{f}$, where Σ^+ is a diagonal matrix of size $N_{coeff} \times N_{coeff}$ whose elements are given by σ_k^{-1} if $\sigma_k \geq \varepsilon \times \sigma_{max}$ (with $\sigma_{max} = \max_k |\sigma_k|$) and 0 otherwise. ε accounts for the relative accuracy of $f(\vec{x})$. For analytical functions, we take $\varepsilon = 10^{-10}$.

Once the quadratic approximation of the fitness has been established, we must determine the solution of $\vec{\nabla} f = 0$ in order to infer the position of the optimum. The solution is given formally by $\vec{x}^* = \vec{x}_{ref} - \Delta A_2^{-1}\vec{A}_1$. Since the matrix A_2 may be non-invertible, we use an approach based on the spectral decomposition of A_2. Since the matrix A_2 is symmetric, its eigensystem $A_2\vec{x}_k = \lambda_k\vec{x}_k$ is characterized by real eigenvalues λ_k and the eigenvectors $\vec{x}_k$ form an orthonormal basis. It is useful at this point to define $\lambda_{max}=\max_k |\lambda_k|$ and $\lambda_{min}=\min_k |\lambda_k|$. The solution of $\vec{\nabla} f = 0$ is then given by

$$\vec{x}^* = \vec{x}_{ref} - \Delta \sum_k \frac{\vec{x}_k.\vec{A}_1}{\lambda_k}\vec{x}_k \qquad (2)$$

where the sum is restricted to the eigenvalues for which $|\lambda_k| \geq \varepsilon_{inv} \times \lambda_{max}$ in order to avoid numerical instabilities. For analytical functions, the best results are obtained with $\varepsilon_{inv} = 10 \times$

$(\lambda_{max}/\lambda_{min}) \times \varepsilon$, where the ratio $\lambda_{max}/\lambda_{min}$ accounts for condition number of the matrix A_2. For problems in which the accuracy of the fitness is limited to three digits, we recommend using $\varepsilon_{inv} = \varepsilon = 10^{-3}$.

If the solution $\vec{x}^*$ provided by this procedure is within the boundaries $[x_i^{min}, x_i^{max}]$ specified for each variable, we inject in the population an individual that encodes for $\vec{x}^*$. We repeat otherwise this procedure up to three times by increasing the half-width parameter W used for the selection of the data points that contribute to this analysis (we consider increments of 2 steps).

REFERENCES

1. Haupt R.L. and Werner D. H. - Genetic Algorithms in Electromagnetics, J. Wiley & Sons: Hoboken, NJ, 2007.
2. Eiben A. E. and Smith J. - Introduction to Evolutionary Computing; Springer-Verlag: Berlin, 2007.
3. Eiben A. E. and Smith J. - From Evolutionary Computation to the Evolution of Things, Nature **521** (2015) 476-482.
4. Chen X., Ong Y. S., Lim M. H. and Tan K. - A Multi-Facet Survey on Memetic Computation, IEEE Transactions on Evolutionary Computation **15** (2011) 591-607.
5. Rasheed K., Ni X. and Vattam S. - Comparison of Methods for Developing Reduced Models for Design Optimization, Soft Computing **9** (2005) 29-37.
6. Regis R. and Shoemaker C. - Local Function Approximation in Evolutionary Algorithms for the Optimization of Costly Functions, IEEE Transactions on Evolutionary Computation **8** (2004) 490-505.
7. Paenke I., Branke J. and Jin Y. - Efficient Search for Robust Solutions by Means of Evolutionary Algorithms and Fitness Approximation, IEEE Transactions on Evolutionary Computation **10** (2006) 405-420.
8. Wanner E., Guimaraes F., Takahashi R. and Fleming P. - Local Search with Quadratic Approximations into Memetic Algorithms for Optimization with Multiple Criteria, Evolutionary Computation **16** (2008) 185-224.
9. Judson R. - In Reviews in Computational Chemistry, K. B. Lipkowitz and D. B. Boyd Ed., VCH Publishers Inc.: New York, 1997; Chapter 10, pp 1-73.
10. Whitley D., An Overview of Evolutionary Algorithms: Practical Issues and Common Pitfalls, Information and Software Technology **43** (2001) 817-831.
11. Mayer A. and Bay A., Optimization by a Genetic Algorithm of the Light-Extraction Efficiency of a GaN Light-Emitting Diode, Journal of Optics **17** (2015) 025002.
12. Rowe J., Whitley D., Barbulescu L. and Watson J.-P., Properties of Gray and Binary Representations, Evolutionary Computation **12** (2004) 47-76.
13. Bhat G. and Savage C., Balanced Gray Codes, Electronic Journal of Combinatorics (1996) **3** R25.
14. Murayama A. and Kanasugi A., A Novel Coding Method for Genetic Algorithms Based on Redundant Binary Numbers, Artificial Life and Robotics (2010) **15** 306-308.
15. Posik P., Huyer W. and Pal L., A Comparison of Global Search Algorithms for Continuous Black Box Optimization, Evolutionary Computation (2012) **20** 509-541.

16. Hansen N. and Ostermeier A., Completely Derandomized Self-Adaptation in Evolution Strategies, Evolutionary Computation (2001) **9** 159-195.
17. Mayer A., Gaouyat L., Nicolay D., Carletti T. and Deparis O., Multi-Objective Genetic Algorithm for the Optimization of a Flat-Plate Solar Thermal Collector, Optics Express (2014) **22** A1641- A1649.
18. Mayer A., Muller J., Herman A. and Deparis O. - Optimized Absorption of Solar Radiations in Nano-Structured Thin Films of Crystalline Silicon via a Genetic Algorithm, Proc. of SPIE (2015), 9546, 95461N-01.
19. Smith J., Evolutionary Genetics, Second Edition; Oxford University Press: Oxford, 1998.
20. Golub G. and Kahan W., Calculating the Singular Values and Pseudo-Inverse of a Matrix, J. Soc. Ind. Appl. Math. Ser. B Numer. Anal. **2** (1965) 205-224.